Attorne[illegible]cket No.: 10559-421001 / P10435

APPLICATION

FOR

UNITED STATES LETTERS PATENT

09872525.053101

TITLE:	IDENTIFYING DATA LOSS IN A TRANSMISSION SYSTEM
APPLICANT:	STUART W. SHERLOCK

CERTIFICATE OF MAILING BY EXPRESS MAIL

Express Mail Label No. ET371085926US

I hereby certify under that this correspondence is being deposited with the United States Postal Service as Express Mail Post Office to Addressee with sufficient postage on the date indicated below and is addressed to the Commissioner for Patents, Washington, D.C. 20231.

5-31-01
Date of Deposit

[illegible]
Signature

Gil Vargas
Typed or Printed Name of Person Signing Certificate

IDENTIFYING DATA LOSS IN A TRANSMISSION SYSTEM

STATEMENT RE COLOR DRAWINGS

The file of this patent contains at least one drawing executed in color. Copies of this patent with color drawings will be provided by the Patent and Trademark Office upon request and payment of the necessary fee.

TECHNICAL FIELD

This invention relates to identifying data loss in a transmission system.

BACKGROUND

Data that is transmitted through a transmission system experiences delays (i.e., latency) between two system end-points, such as a transmitter and a receiver. The data also experiences jitter, which can contribute to delay between end-points. Latency and jitter can result in loss of data.

Data loss is particularly problematic in audio applications. For example, in IP (Internet Protocol) telephony significant data loss can adversely affect resulting audio.

DESCRIPTION OF THE DRAWINGS

Fig. 1 is block diagram of a transmission system.

Fig. 2 is a flowchart showing a process for identifying data loss in the transmission system of Fig. 1.

Fig. 3 (color) is a plot of transmitted and received waveforms from the transmission system of Fig. 1.

Fig. 4 is a plot showing data differences between the transmitted and received waveforms of Fig. 3.

Fig. 5 is a plot showing data loss from various waveforms generated by the process of Fig. 2.

Fig. 6 (color) is a plot of shifted transmitted and received waveforms from the transmission system of Fig. 1.

Fig. 7 is a plot showing data differences between the transmitted and received waveforms of Fig. 6.

Fig. 8 (color) is a plot of further shifted transmitted and received waveforms from the transmission system of Fig. 1.

Fig. 9 is a plot showing data differences between the transmitted and received waveforms of Fig. 8.

Fig. 10 is a block diagram of a computer on which the process of Fig. 2 may be implemented.

Int a'>

DESCRIPTION

Fig. 1 shows a transmission system 10 for transmitting data between two, or more, end-points. Transmission system 10 includes a transmitter 12, a transmission medium 14, and a receiver 16. In this embodiment, transmission medium 14 is a network medium, such as Ethernet; although other types of media may be used. Transmitter 12 and receiver 16 are adapted to transmit and receive, respectively, audio data, such as data for IP telephony. It is noted, however, that other types of data, both audio and non-audio, may be transmitted over transmission system 10 and processed as described below.

Receiver 16 includes buffer(s) 18 for receiving, and buffering, audio data packet(s) prior to playback. These buffer(s) include some type of circuitry for coding/decoding (CODEC) the audio data, if necessary. For example, the buffers may include a G.723 CODEC or an MPEG (Motion Picture Experts Group) CODEC.

Delays typically occur in transmission from transmitter 12 to receiver 16. The delays may occur in the receiver or transmitter themselves, and/or in transmission medium 14. Jitter is known to have significant influence in the amount of delay in the system. The "system-wide" delay adversely affects the quality of the received audio. The delay is referred to as "system-wide" because it encompasses delay from transmitter 12, delay in receiver 16, and jitter in receiver 16, not just delay from transmission medium 14.

To increase the quality of the received audio, the amount of delay may be reduced and the size of the receiver buffer may be adjusted. Fig. 2 shows a process 20 for detecting the amount of delay in transmission system 10. Process 20 may be performed to test transmission system 10 using "test" audio data, before "real" audio data is transmitted over transmission system 10. Once the amount of delay has been detected, post-processing may be performed to compensate for this delay. The compensation may include varying the receiver buffer size and the amount of traffic permitted on the transmission system. Process 20 may be performed in receiver 16 or in an external computer (not shown in Fig. 1) that can receive data from transmission system 10.

Process 20 obtains (22) a transmitted waveform from transmitter 12 and a received waveform from receiver 16. In this context, the transmitted waveform is the signal that transmitter 12 transmits to receiver 16 over transmission medium 14. The received waveform is the signal that receiver 16 receives from transmitter 12 over transmission medium 14.

Process 20 normalizes (24) both the transmitted and received waveforms so that they both contain only positive data. The transmitted and received waveforms are initially aligned as best as possible. Process 20 compares (26) the transmitted and received waveforms. The comparison is performed by superimposing the transmitted waveform 28 and the received waveform 30 (Fig. 3). In the plot of Fig. 3, the transmitted and received waveforms are superimposed with the best possible (at this point) alignment of the received waveform to the transmitted waveform, as described below.

Process 20 determines (32) the difference between the transmitted waveform 28 and the received waveform 30. For the plot shown in Fig. 3, the difference between the transmitted and received waveforms is shown in the plot of Fig. 4. At this point, process 20 may divide the difference data of Fig. 4 into groups of "envelopes". In this context, an envelope is a portion (e.g., 25%) of the data buffer in the receiver. Process 20 determines and stores (34) the data loss and loss rate (i.e., the percentage of data lost from the transmitted to the received waveform) for each envelope of data.

The "unshifted" comparison of the received waveform 30 and the transmitted waveform 28 constitutes the initial ("0th") increment shift. As described in more detail below, process

20 shifts the received waveform relative to the transmitted waveform (or vice versa) in order to find the amount of shift at which the data loss rate is relatively low (e.g., the lowest data loss). Received waveform 30 may be shifted relative to transmitted waveform 28 or, alternatively, transmitted waveform 28 may be shifted relative to received waveform 30. For each shift in the -X direction, there is an equal and corresponding shift of the same waveform in the +X direction. Thus, the number and size of shifts in each direction (e.g., +/-X) is the same. Each pair of shifts constitutes a shift group, such that a shift group (except for the 0th shift noted above) has a positive (+X) component and a negative (-X) component. Thus, an odd number of shift points will generally be obtained by process 20. Generally, two pairs of shifts plus the initial shift (for a total of five shift groups) is sufficient to determine a vertex or if added shifting is needed. As described below with respect to Fig. 5, the shifts are plotted against the data loss rate to determine which shift produces the least data loss.

Referring back to Fig. 2, process 20 determines (36) if the desired number of shifts have occurred. The number of shifts in one direction (e.g., -X) may be determined based on the size of a CODEC buffer in receiver 16. Table 1 (below) shows examples of shift group sizes for three well-known CODEC buffers (G.723, G.711 and G.729) used in audio receivers. The shift group sizes of Table 1 are provided both in terms of data samples and milliseconds (msec).

Table 1

CODEC Buffer	Buffer Size In msec	Shift Group Size In msec	Shift Group Size In # Of Data Samples
G.723	30	5	10
G.711	120	20	40
G.729	10	2	4

Thus, for the G.723 CODEC buffer, each shift in a shift group is 5 msec covering 10 data samples. Since the CODEC buffer size is 30 msec, there are a total of 6 (=30÷5) shift groups, resulting in a total of 6 shifts in the -X direction and 6 shifts in the +X direction. Similar calculations can be used to obtain the number of shift groups/shifts sizes for each of the other buffers noted in Table 1. Alternatively, the size and number of the shift groups may be set as desired and may or may not be a multiple of the data stored in the buffers.

The minimum number of shift groups for a buffer in receiver 16 is determined beforehand and may be stored, e.g., in a memory of a machine executing process 20. Referring to Fig. 2, if the desired number of shifts have not occurred (which would generally not be the case immediately following the 0th shift), process 20 shifts (38) the received waveform relative to the transmitted waveform. In this representative embodiment, received waveform 30 (Fig. 3) is shifted in the -X direction relative to transmitted waveform 28; however, as noted above, transmitted waveform 28 may be shifted instead. Received waveform 30 is shifted because that is the waveform that is delayed through transmission system 10.

Fig. 6 shows received waveform 30 shifted relative to transmitted waveform 28. Process 20 compares (26) determines (32) the difference in data after shifting received waveform 30 relative to transmitted waveform 28. This difference is plotted in Fig. 7. Process 20 stores (34) the data loss rate, which corresponds to this difference in, e.g., an array in memory. Process 20 determines (36) if the desired number of shifts has occurred. Assuming that this is not the case, process 20 shifts (38) the transmitted waveform again - this time in the equal and opposite direction as the previous shift in the current shift group, i.e., in the +X direction. Process 20 repeats blocks 26, 32, 34, 36, 38 of Fig. 2 until the number of shifts of waveforms has occurred for the current buffer size and the corresponding data loss rates for the shifts have been stored.

One of the foregoing shifts may result in a minimum (or, at least, reduced) data loss rate between the two waveforms. The shift that produces this effect in the current example is shown in Fig. 8, with the data loss rate for that shift being plotted in Fig. 9. Process 20 identifies the shift that produces a minimum difference by comparing the data loss rates stored in the array in memory. In this embodiment, process 20 plots (40) the data loss rates (i.e., differences in data) for each shift in each increment shift group. Fig. 5 shows the plot 42 generated for this example. Process 20 identifies (41) a lowest vertex point 44 on the plot. The lowest vertex point 44 corresponds to the shift at which the least amount of data is lost between the transmitted and received waveforms. The least data loss constitutes the smallest difference in data between the transmitted and received waveforms.

If a vertex in the plot has not been reached (43) (e.g., if the plot is an increasing function and not parabolic), process 20 increments (45) the number of shift groups. That is, process 20 adds two shift groups, resulting in two more shifts in both the -X and +X directions. Process 20 then returns to 36, whereafter the process continues until a lowest vertex point is obtained.

As shown in Fig. 5, the shift that corresponds to the lowest vertex point 44 on plot 42 is the "-3" shift, i.e., the shift in the -X direction for the third shift group. Knowing that this shift produces the minimum difference, receiver 16 can be programmed to implement this shift on all "non-test" audio data that it receives from transmitter 12 over transmission medium 14. By implementing this shift, the amount of data loss in the audio data can be reduced, thus providing audio with less noise and higher fidelity.

Fig. 10 shows a computer 46 on which process 20 may be implemented. Computer 46 includes a processor 48, a memory 50, and a storage medium 52 (see view 54). Storage medium 52 contains a buffer for storing received "test" audio data 56 and machine-executable instructions 58 that are executed by processor 48 out of memory 50 to perform process 20 on data 56. The components of computer 46 may be located in receiver 16 in order to perform process 20 in receiver 16.

Process 20, however, is not limited to use with the hardware and software of Fig. 10. It may find applicability in any computing or processing environment. Process 20 may be implemented in hardware, software, or a combination of the two. Process 20 may be implemented in computer programs executing on programmable computers or other machines that each include a processor, a storage medium readable by the processor (including volatile and non-volatile memory and/or storage components), at least one input device, and one or more output devices. Program code may be applied to data entered using an input device (e.g., a mouse or keyboard) to perform process 20 and to generate output information.

Each such program may be implemented in a high level procedural or object-oriented programming language to communicate with a computer system. However, the programs can be implemented in assembly or machine language. The language may be a compiled or an interpreted language.

Each computer program may be stored on a storage medium/ article (e.g., CD-ROM, hard disk, or magnetic diskette) that is readable by a general or special purpose programmable computer for configuring and operating the computer when the storage medium or device is read by the computer to perform process 20. Process 20 may also be implemented as a machine-readable storage medium, configured with a computer program, where, upon execution, instructions in the computer program cause a machine to operate in accordance with process 20.

The invention is not limited to the specific embodiments described above. For example, process 20 is not limited to use with audio data. It can be used with video or any other type of streaming data where alignment and time considerations affect the quality of the resulting output. Process 20 is not limited to use with any type of transmission medium or network. For example, process 20 may be used with ATM (Asynchronous Transfer Mode) networks, LANs (Local Area Networks), WANs (Wide Area Networks), or any other network. Process 20 is not limited to use with audio transmitters and receivers (or computers) such as those shown in the figures. It may be performed on any one or more machines having access to transmitted and received waveforms.

The blocks shown in Fig. 2 merely represent one way that process 20 may be ordered. The blocks may be reordered, if desired, to perform the same function. For example, process 20 can be extended to account for a case where a lowest vertex is not identified given the desired number of shifts. In this case, for example, the number of shifts/shift groups may be adjusted until a lowest vertex is identified. That is, additional data (e.g., an additional data envelope) may be obtained from the buffer and additional shift groups (e.g., two at a time) may be added until the lowest vertex point is identified for the two waveforms. Knowledge of the CODEC buffer size is helpful in determining the shift sizes.

The attached Appendix shows pseudo-code for implementing one embodiment of process 20.

Other embodiments not described herein are also within the scope of the following claims.

What is claimed is: